

VALENCE

Design Document

IMD 4901A

Supervisor: Prof. Chris Joslin

October 03, 2015

Authored by:

Team NeonPanda

Members:

Jake Deugo

Tyler Dinardo

Laryssa Ribeiro Da Silva

Weri Sin

Zachary Sullivan

Vishesh Thanki

Version	Date	Summary
1	October 9th, 2015	Initial Design Document
2	December 31st, 2015	Final Design Document

Scope

This document outlines the full player experience and design for *Valence*, an RTS survival game developed by *Neon Panda*.

Table of Contents

1. Project Overview

1.1 Team Members

1.2 Pre-Gameplay Experience

1.2.1 Website

1.3 Gameplay Overview

1.4 Story and Setting

1.5 Factions

1.5.1 Folk

1.5.2 Folk Style

1.5.3 Elite

1.5.4 Elite Style

2. Game Dynamics

2.1 Resource Management and Currency

2.2 Settlement Building

2.3 Exploration

2.5 Agent Logic

2.5.1 Agent Attributes

2.5.2 Agent State Machines

2.5.3 Agent Pathfinding

2.5.4 Dynamic Agent Needs

2.5.5 Agent Social Physics

2.6 Gameplay Inspiration and References

2.6.1 Simcity

2.6.2 Banished

2.6.3 Fallout Shelter

2.6.4 XCOM: Enemy Unknown

2.6.5 Fire Emblem

3. Controls and Interface

3.1 Build Mode Control

3.2 Explore Mode Control

3.3 Icons

3.4 Interface Reference

3.4.1 Banished

3.4.2 Starcraft II

4. Game Environment

4.1 Metro Stations

4.1.1 Main Station

4.1.2 Exploration Stations

4.2 Buildings

4.3 Props

5. Technology

5.1 Game Engine

5.2 Folk Generation

5.3 Procedural Prop Placement

6. Style

6.1 Art

6.2 Art Influences

6.2.1 XCOM

6.2.2 Metro 2033

6.3 Sound

6.3.1 Grow Home

6.3.2 Pandemic 2

6.3.3 Superhot

1. Project Overview

Team *NeonPanda* is developing a city building game that aims to implement an advanced state-of-the-art agent modelling platform **that captures social interaction amongst** artificially intelligent characters.

The game titled, *Valence*, will task players to maintain and manage a colony of asylum seeking characters (referred to as agents), who have sought refuge in an abandoned urban metro system. *Valence* is a single-player experience built for PC. *Valence* will place players in a God like perspective, overseeing the development and resource management of their colony. Features contained within *Valence* are influenced heavily by its unique narrative (see 1.2 Story Synopsis). Alongside its city building component, *Valence* will incorporate a world exploration system. Within this system, players will be tasked to select, strategize, and execute agent behavior through a series of turn-based combat scenarios, rewarding successful outcomes with loot items which can further improve agent attributes.

1.1 Team Members

Team *NeonPanda* consists of 6 group members with a key designated role within the project.

- **Laryssa:** Character Artist | Email: laryssa.silva@carleton.ca
- **Tyler:** Gameplay Programmer | Email: tyler.dinardo@carleton.ca
- **Jake:** Environment Designer | Email: jake.deugo@carleton.ca
- **Weri:** Building Artist | Email: weri.sin@carleton.ca
- **Zachary:** Systems Programmer | Email: zachery.sullivan@carleton.ca
- **Vishesh:** User Experience Developer | Email: vishesh.thanki@carleton.ca

1.2 Pre-Gameplay Player Experience

In order to play the game, players must download the game executable through the game's official website.

1.2.1. Website

The website will be created using HTML, CSS, and Javascript. It will contain the following sections:

- **Development Blog:** This section will update players on the different parts of the game's development, which version it is presently in, and the date it will be released.
- **Download:** The section where players may download the game.
- **About:** This section will have information on the team, and the members. This section will also discuss the main storyline of the game, and it will contain a description of each neighbourhood child. Furthermore System requirements to play the game will be included.
- **Media:** This section will consist of wallpapers, screenshots, and other media for the game.
- **The Team:** This section will include info regarding the team and ways to contact them.

1.3 Gameplay Overview

Upon launch of the game, players are introduced to a short narrative description showcasing and explaining why the player is placed into this environment. Once the game has fully loaded, players will be placed into a blank canvas environment. Players must immediately begin planning the layout of their colony, as well as how to use their limited starting population to its peak efficiency and assigning roles to them as buildings are created.

Players must choose from a suite of starting buildings to construct (each with their own unique purpose and requirements). These starting structures include, a housing unit for a settler to live in, a farm to produce sustainable food, a generator to provide power to the colony (and light the environment), and a storage area to hold all resources produced and collected.

1.4 Story and Setting

Valence takes place in the year 2049 within the subterranean post bio-apocalyptic world of Montréal's extensive Métro subway. A world where the accelerated onset of global warming and pollutant clouds of airborne chemical and biological pathogens continue to mutate, ravishing the atmosphere and the planet's surface (known solely as "The Event"). While society and its economy continues to sputter along barely functional, it has in reality, split into two distinct factions the Folk and the Elite. Each of whom struggle over fair democratic access to the remaining resources.

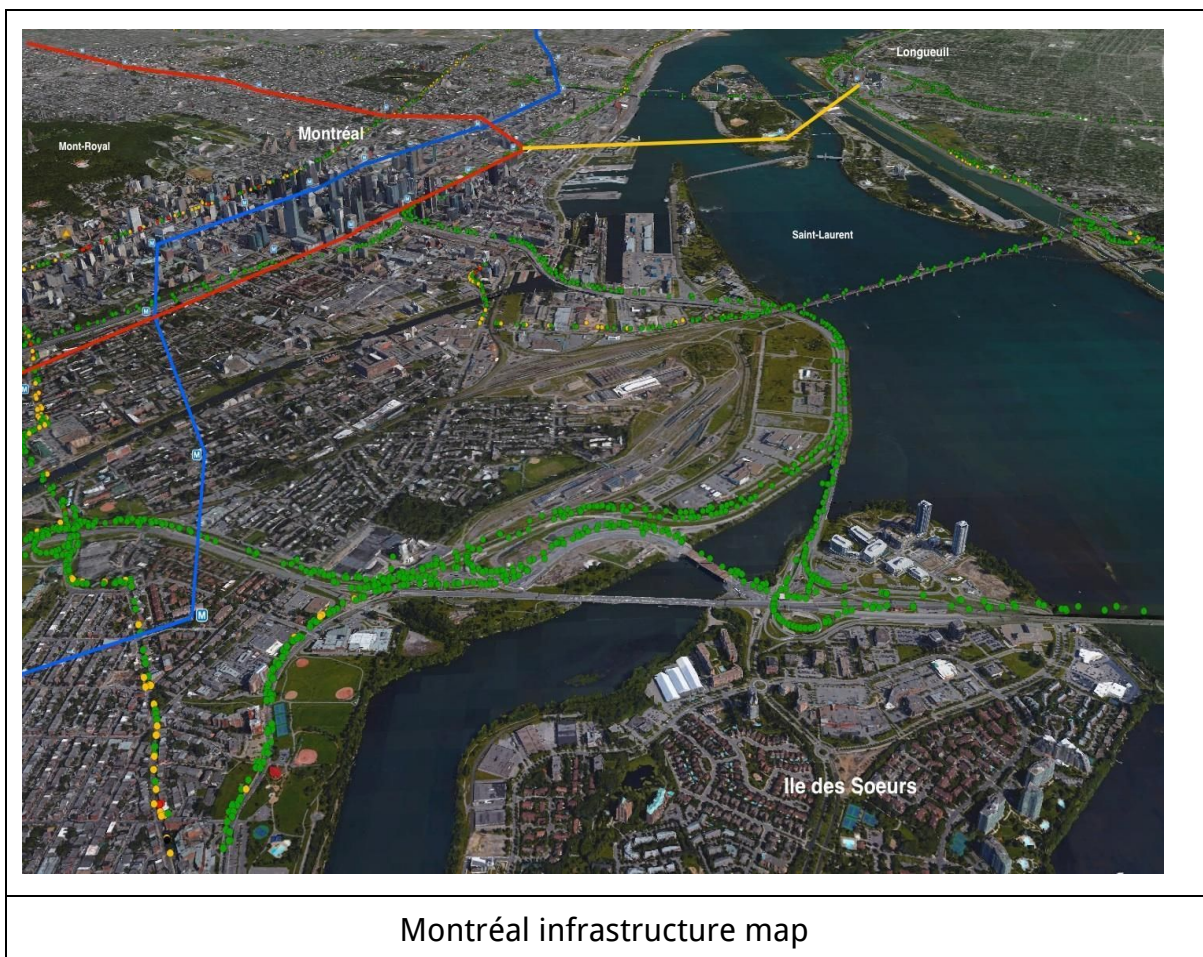
1.5 Factions

Valence features two distinct factions, the Folk and the Elite. Both groups are locked in a constant struggle to gain control over Montréal's last remaining resources.

1.5.1 Folk

Made up from a majority of the original population of Montréal, *The Folk*, are resilient people from all walks-of-life. Skilled in engineering, communications and

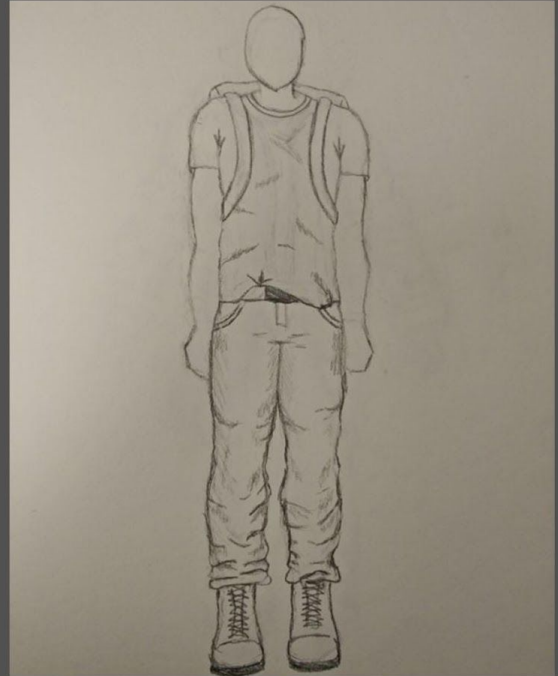
administration, *The Folk*, were slow after *The Event* to collectively understand its implications, organize into groups and plan a course of action. Consequently, they have been forced to seek underground asylum in the pollution free Métro subway. This also allows them to travel easily throughout most of Montréal's core as well as control much of the city's infrastructure underbelly which they have managed to largely keep operational through numerous innovations. *The Folk* continue to use this latter strength to leverage food and energy concessions from their struggle with *The Elite*.



1.5.2 Folk Style

The style of the folk faction needs to portray the environmental circumstances in which they are in and within what faction they are in. Since the folk faction has limited resources, they have rugged and coarse style of clothing. They also need

adequate equipment to keep their resources. For this reason folk faction has a refugee look. Below are some concept art for female and male folk clothing.



Folk Male

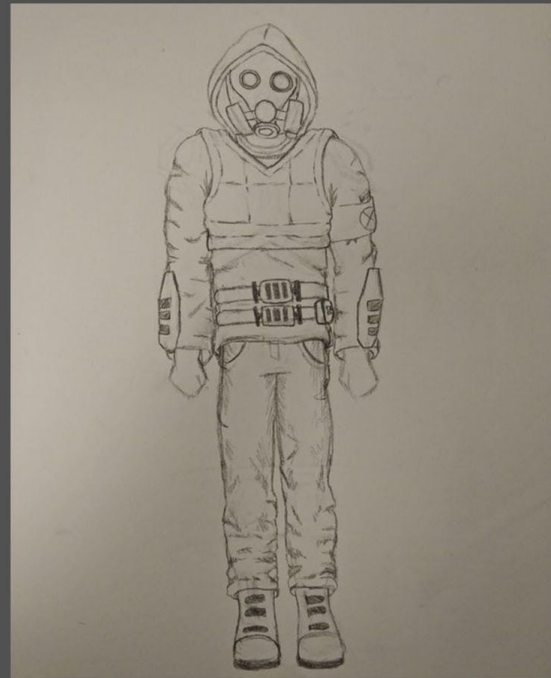
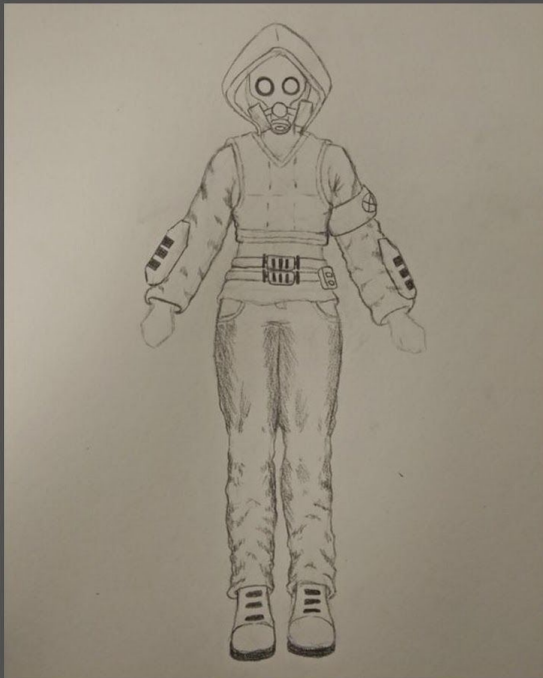


1.5.3 Elite

The Elite are the upper echelon of Montréal leading lights, 'captains of industry & finance', police, and political leaders prior to The Event. The Elite were quick to take advantage of The Event by commandeering control over most of the city's pre-existing resources (e.g. money, power transmission lines, surface road & rail networks, shipping terminals that provides access to imported resources from less polluted regions, ex-urban farms and food depots adjacent to Montréal through the south shore town of Longueuil). These resource however, while critical to both factions, are slowly decaying due to the relentless onslaught of The Events pollutant fallout. The Elite have not adapted well to this new reality, proving themselves to be increasingly unimaginative, repressive and authoritarian toward *The Folk* in order to compel their infrastructure support. Consequently, The Elite have isolated themselves from The Folk principally in downtown office towers and on the secure residential island of Ile des Soeurs (Nun's Island)

1.5.4 Style

Elite faction contain most of the resources on the metro, consequently they dominate the Folk faction. The style of the elite represent their class and their power over the folk. Having possession of most of the available resources, the elite have superior equipment and armour on them. Having gas masks gives the elite the ability to go above ground for a certain amount of time. The elite will be set in the explore mode in which they will “fight” against the folk to retain their resources. The elite faction will consist in having a uniform since they represent order and control. Below are the concept art for female and male Elite clothing.



Elite Characters

2. Game Dynamics

The core to the dynamics of *Valence* are both a building and exploration system. The combination of these mechanics have been heavily inspired by games such as Simcity, XCOM, Banished, and City Skylines.

Since players will be managing a colony of settlers who have been forced to live within these harsh environments, strategy, planning and micromanagement play a key role in how successful players will be at sustaining and growing an impoverished population.

As players progress further into the game, they will find themselves pushed to venture deep within the mysterious expanses of the dilapidated metro system. Players will encounter powerful enemies, engaging in combat with the hopes of obtaining scrap and other loot to further increase the attributes of their settlers. If successful, players will be able to marvel in the glory of their wonderful creation, yet if failure strikes and their population sputters away, players will be forced to begin a new.

2.1 Resource Management and Currency

In order to make the game as easy to learn and play as possible, *Valence* incorporates a single currency and three primary resources:

- **Scrap (currency):** Scrap is required to construct different buildings. Different buildings come with a price per square unit. Scrap is acquired through the Exploration Mode.
- **Food (resource):** What keeps your settlers alive! Food is acquired from farms. Settlers consume food throughout the day, meaning players must have a constant production of food to meet the needs of the settlement population.
- **Power (resource):** Like food for Settlers, Power is what keeps your existing zones operating. Power is produced from a Power Station. Power acts as a recurring maintenance fee to keep your zones working. If your current

Power production fails to meet the settlement needs, buildings will become non-operational until power is restored to them.

- **Water (resource):** Water is a central resource for survival, and contaminated Water can have a detrimental effect on the population. Water is generated from a Water Purification zone. Insufficient water production would lead to Settler illness, which negatively affects Settler health and morale.

2.2 Settlement Building

As the primary gameplay mode, **Build mode** is where users can craft their own settlement. If a player wants to develop their settlement, they must first select a zone type and build it on the game grid. If the player has sufficient scrap, the zone will be built. The larger the specified area, the more expensive the zone will be to develop. Scrap is acquired through the Exploration Mode. **[Refer to Build Mode Flowchart]**

Table 1 - Zone Types

Zone Type	Function	Average Scrap Cost	Role
Shelter	Allows for population growth	Low	N/A
Farm	Food Production	Low	Farmer
Power Station	Power Production	Low	Mechanic
Water Purification	Water Production	Low	Technician
Storage	Increases Max Resource Limit	Medium	Factory Worker
Medical	Heals wounded Settlers	Medium	Doctor
School	Develops Settler Attributes	High	Teacher
Shrine	Increases Settlement Morale	High	Preacher
Tavern	Increases Settlement Morale	High	Bartender

Settlers must be assigned to fulfill a role associated with each building, or else it will not serve its designated function. Each role is tied to a particular settler

attribute, and settlers with a high level in that particular attribute will be most proficient in the role. For example: the attribute associated with the Farmer role is strength and as such settlers with high Strength attributes are best suited to that role.

Table 2 - Agent Roles

Role Name	Primary Attribute Modifier	Associated Zone
Farmer	Strength	Farm
Mechanic	Perception	Power Plant
Technician	Perception	Water Purification
Storage Worker	Agility	Storage Area
Doctor	Intelligence	Medical Area
Teacher	Intelligence	School
Preacher	Charisma	Shrine
Bartender	Charisma	Tavern

2.3 Exploration

Explore Mode is the tactical gameplay layer of *Valence*. It offers a more immediate and reactive gameplay experience. The exploration mode takes the form of a high stakes turn-based strategy game set on a grid. Players move their party of Settlers along the map, attempt to secure resources and fight back against the Elite.

Objectives

Secure the scrap guarded by the Elite encampment.

Win Conditions

- Secure Enemy Scrap, and return it to the 'retreat' square
- Eliminate all Enemy units

Lose Conditions

- All Folk Units are Eliminated

Draw Conditions

- Player returns to 'retreat' square and opts out of scenario
- Time Limit is reached (level specified)

Turn Structure

Explore Mode operates on a basic turn structure. The player always takes their turn first. During a turn a player can give orders to any and all of their available units, in the form of 'actions' (see section 3.0 for a complete list of player actions). At any times the game can end given that one of the Win/Lose/Draw conditions are met, regardless of whose turn it is.

Player Turn

During the player turn players can give orders to their units in the form of 'actions'.

Any number of units can be given orders each turn, however each unit can only conduct 1 movement action, and one other action.

Enemy Turn

During the enemy turn the enemy units follow a similar structure to that of player units. If an enemy has not detected a player unit the enemy unit will continue whatever its idle action is (either patrolling between 2 points OR standing still). For more detailed information on Enemy Unit actions when a player unit has been detected see the section on Player Detection.

Actions

Move - The unit moves from a their current space to another within their movement range. By default a Folk unit can move 6 spaces. For every additional point a unit has in Agility they gain an additional movement space. If a folk unit moves half its total movement range or less, it is considered to

be 'sneaking' and does not emit any sound [more on this in Player Detection].

Attack - If one or more Elite are within a Folk unit's 'Attack' Range, a Folk unit can be ordered to attack that Elite. A unit's Attack Range is governed by their equipped weapon. By default all folk units are equipped with 'Fists' which have an attack range of 1. A chance to hit is calculated based upon the equipped weapon's accuracy rating. If a hit is landed damage is dealt. Damage is dealt bases upon a weapon's power rating, 1 additional point of damage is applied for every point an Agent has in Strength.

Wait - A player issues no further commands to a unit.

Collect Scrap - If a unit is within 1 unit of the scrap, a unit can be ordered to collect the scrap. While a unit is holding the scrap they are no longer able to attack.

Drop Scrap - If a unit is holding the scrap, they can opt to drop the scrap. This counts as an action and as such they cannot attack on the turn in which they drop the scrap.

Retreat - If a unit is atop a 'Retreat' tile, players can opt to retreat. When a player opts to retreat the scenario ends, and any units outside the 'retreat' range are lost.

Complete Mission - If a unit is atop a 'Retreat' tile, and is in possession of the Scrap, players can opt to complete the mission, thus fulfilling scenario win condition A. When a player opts to complete the mission the scenario ends, and any units outside the 'retreat' range are lost.

Player Detection

The Enemy Elite function akin to that of the Folk Units. However the Elite are not given an omnipotent view of the battlefield. Enemies have two methods of detecting Folk Units. An audio perimeter, and a vision cone.

Elite have an audio detection radius of 3 units in any direction. Any action that a folk conducts within radius will alert an Elite Unit to their presence. When detected the Elite Unit's facing is directed at the Folk. If after facing the direction of the sound the Elite Unit still cannot detect the folk unit, a Pursuit Phase is initiated and the Elite Unit will navigate to the source of the sound.

Elite have a vision cone that extends 6 units in front of them, or until an obstruction, in a 90 degree arc. If a folk unit navigates into this field of view, the Elite will enter an Combat Phase and will be aware of the Folk Unit's position. If a folk unit manages to navigate outside of all Elite unit's field of view, a last known position is placed at the last spot within his vision cone, the Combat Phase will end and the Elite will enter a Pursuit Phase. Once in a pursuit phase an Elite will navigate to that position.

Enemy Unit Action Priority Chart

Priority	Condition	Action
1	Multiple Player Units are within Attack/Movement Range, and A's health is \leq Enemy Attack	Move and Attack Player Unit A.
2	Multiple Player Units are within Attack/Movement Range, and Player Unit A is in possession of the Scrap	Move and Attack Player Unit A
3	Multiple Player Unit's are within Attack/Movement Range, and Player Unit A has the least health of all player units within range.	Move and Attack Player Unit A

4	Multiple Player Unit's are within Attack/Movement Range, and 2 or more units have the least health of all player units within range.	Move and Attack Player Unit closest to Position.
5	A Single Player Unit 'A' is within Attack/Movement Range.	Move and Attack Player Unit A
6	No Player Units are within Attack/Movement Range. However multiple positions of interest are known.	Navigate to most recently perceived position of interest.
7	No Player Units are within Attack/Movement Range. However a position of interest is known.	Navigate to most recently perceived position of interest.
8	No Player Units are within Attack/Movement Range. No Positions of Interest Exist. Enemy unit is currently not in default patrol route.	Navigate back to default patrol route.
9	No Player Units are within Attack/Movement Range. No Positions of Interest Exist. Enemy unit is currently in default patrol route.	Conduct default patrol route

Weapons

Range: The distance at which a player can initiate an attack with given weapon

Acc: The chance to hit, scale is 0-1

Dmg: Damage dealt upon hit

SND: Effective area of sound created by attacking with this weapon

DROP: Chance of receiving this weapon after a mission.

Modifier: Additional Attributes.

Name	RANGE	ACC	DMG	SND	DROP	Modifier
Fists	1	0.66	1	0	NA	Silent
Baseball Bat	2	0.66	2	0	0.88	Silent
Pistol	6	0.50	4	6	0.56	---
Shotgun	3	0.50	8	12	0.33	---
Rifle	9	0.33	6	12	0.17	---

Rewards

Following a successful scenario players have the opportunity to receive rewards in the form of new Folk, Weapons and Scrap.

2.5 Advanced Agent Logic

Valence leverages a modern artificial intelligence to affect a diverse suite of fluid behaviours by each agent within the game environment. Fundamentally, an agent's behaviours are composed of layered (augmented¹) Finite State Machines (aFSM) which activate or inhibit each other through Brooksian Subsumption² design principles.

Each aFSM behaviour layer shares common access to the "internal conditions" implemented within each agent. An agent "perceives" the environment around them through these "internal conditions", in which specific thresholds act like sensors. Once an agent's "beliefs" have been established, the plan-of-action embodied within an activated behaviour layer's aFSM represents an agent's "desires" (i.e. a resolved path along the game map) with the specific actions enacted by a given aFSM representing the agent's "intentions" (i.e. move to destination at coordinates x, y, z). Collectively, this information represents the motivational state of an agent as well as capturing the deliberative component of the system.

Built upon a BDI architectural framework, agent logic is based on its three core aspects:

Belief - is represented within *Valence* as a preconfigured threshold bound to one or more internal conditions of an agent which when triggered either *activates* or *inhibits* other behavioural actions;

¹ Augmented FSMs use time to gradually decay a behaviour's triggering condition

² MIT's Rodney Brooks defined his Behavioural Subsumption as a way to compose specific behavioral actions into a composite whole that can fluidly switch back and forth between them.

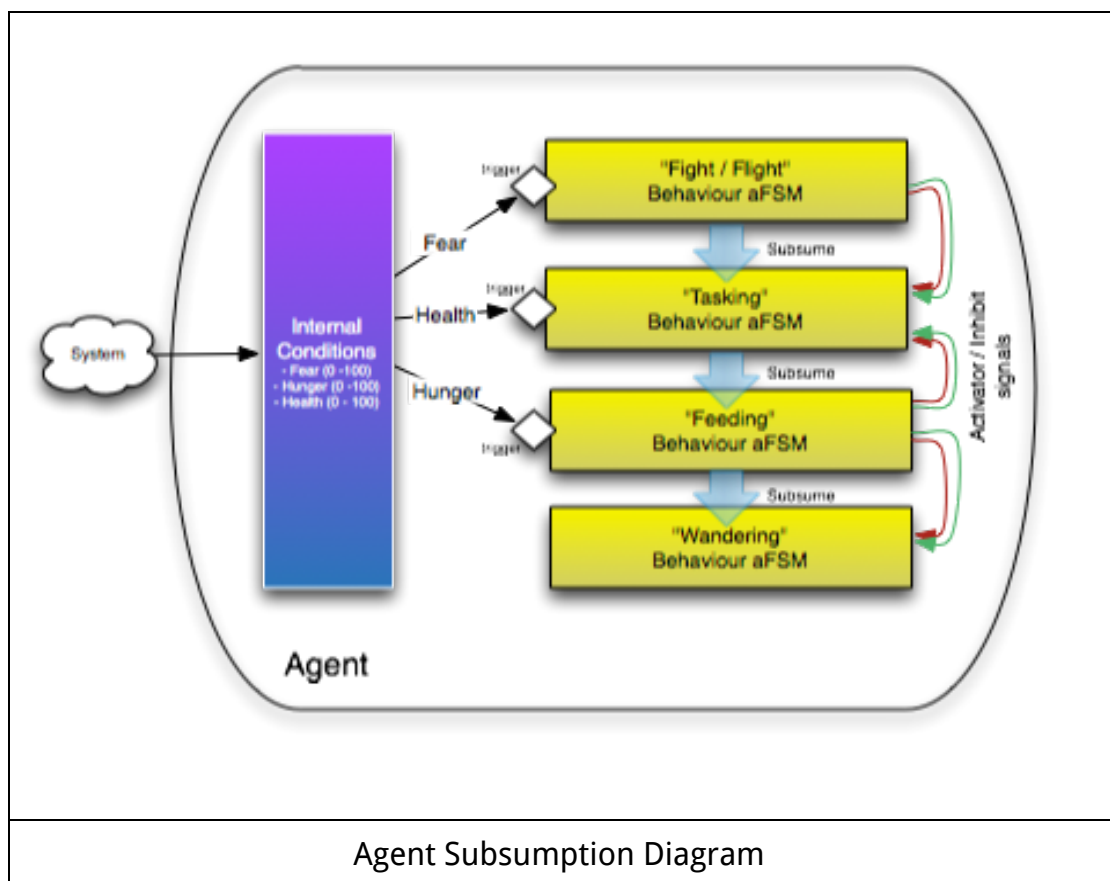
Desire - is represented within *Valence* as the plan-of-action embodied within each activated behavioural layer (i.e. when Wandering it is the path acquired from Unity's pathfinder)

Intent - is represented within *Valence* as the specific actions when executing a particular aFSM (i.e. when Wandering along a given path, move to destination x, y, z)

Consider the following as baseline *Valence* behavioural layers:

- **Wandering** - this lowest priority behavioral layer accounts for an Agent's default behaviour - which is a "*wandering*" action that leverages Unity's pathfinding obstacle avoidance to discover resources within a map zone;
- **Feeding** - The next highest priority behaviour layer "subsumes" the "*wandering*" layer below it - which when triggered by a *<hi-hunger>* threshold (i.e. 70%) - activates the substitution of "*wandering*" destinations with the nearest food source as its preferred destination goal while inhibiting other behavior layers. Once the *<lo-hunger>* threshold triggers (i.e. 30%), it inhibits the "*wandering*" behaviour while activating all other higher behaviour layers should their triggering conditions warrant;
- **Tasking** - This next highest behavioral layer subsumes both the "*wandering*" and "*feeding*" behaviours and is triggered when an agent's *<health>* condition is high and its *<hunger>* and *<fear>* conditions are low, allowing it to pursue the construction or maintenance task assigned to it by the player. Should the *<hi-hunger>* threshold trigger it will inhibit the "*tasking*" behaviour allowing the "*feeding*" behaviour to re-activate;
- **Fight or Flight** - this is the highest behavioral layer and it subsumes all other the lower priority behaviours when it is triggered by a high *<fear>* condition regardless of other conditions. Once this behavioural layer's aFSM has been activated and the "*tasking*" layer inhibited, should either the agent's *<health>*

and/or <hunger> conditions prove to be low then a “flight” reaction occurs moving away from any threat to either a food source or wandering to a safer region of the map zone as the corresponding conditions may dictate. Otherwise, a “fight” reaction occurs engaging the threat directly in combat, during this mode should either the <health> and/or <hunger> conditions diminish to a low state then the agent will revert to a “flight” reaction disengaging from combat.



2.5.1 Agent Attributes

All agents within *Valence* will have a suite of attributes associated with them. Agent attributes are used to emphasize the role assignment mechanic mentioned in section 2.2 Settlement Building.

In addition, agent attributes are a core aspect of each agent's internal logic. As outlined in table 3 below, each attribute influences how an agent will behave.

Table 3 - Agent Attributes

Agent Attribute	Description
Strength	<ul style="list-style-type: none"> - Affects the total amount of resources an agent can transport - Within combat, Strength increases the maximum amount of damage dealt to enemies
Perception	<ul style="list-style-type: none"> - Influences the probability an agent will act based on his needs. <ul style="list-style-type: none"> - Eg. agent would have a higher probability to search for food earlier if their perception value was greater than the default. - Within the exploration mode, agent accuracy during combat is improved. <ul style="list-style-type: none"> - This means that agents who are more accurate, have a greater chance of inflicting damage when engaging an enemy.
Agility	<ul style="list-style-type: none"> - Affects the speed at which an agent can transport a resource from point A to B. <ul style="list-style-type: none"> - Eg. When an agent transports a produce from a farm to a storage building, the transport speed would be faster (If agility was greater than the default value). - Within the exploration mode, agility influences the chance to avoid an attack, as well as the maximum distance an agent can move in combat
Intelligence	<ul style="list-style-type: none"> - Intelligence influences the amount of resources an agent will produce - Within the exploration mode, intelligence increases an agent's chance to inflict a critical hit (double the maximum damage output of the agent)

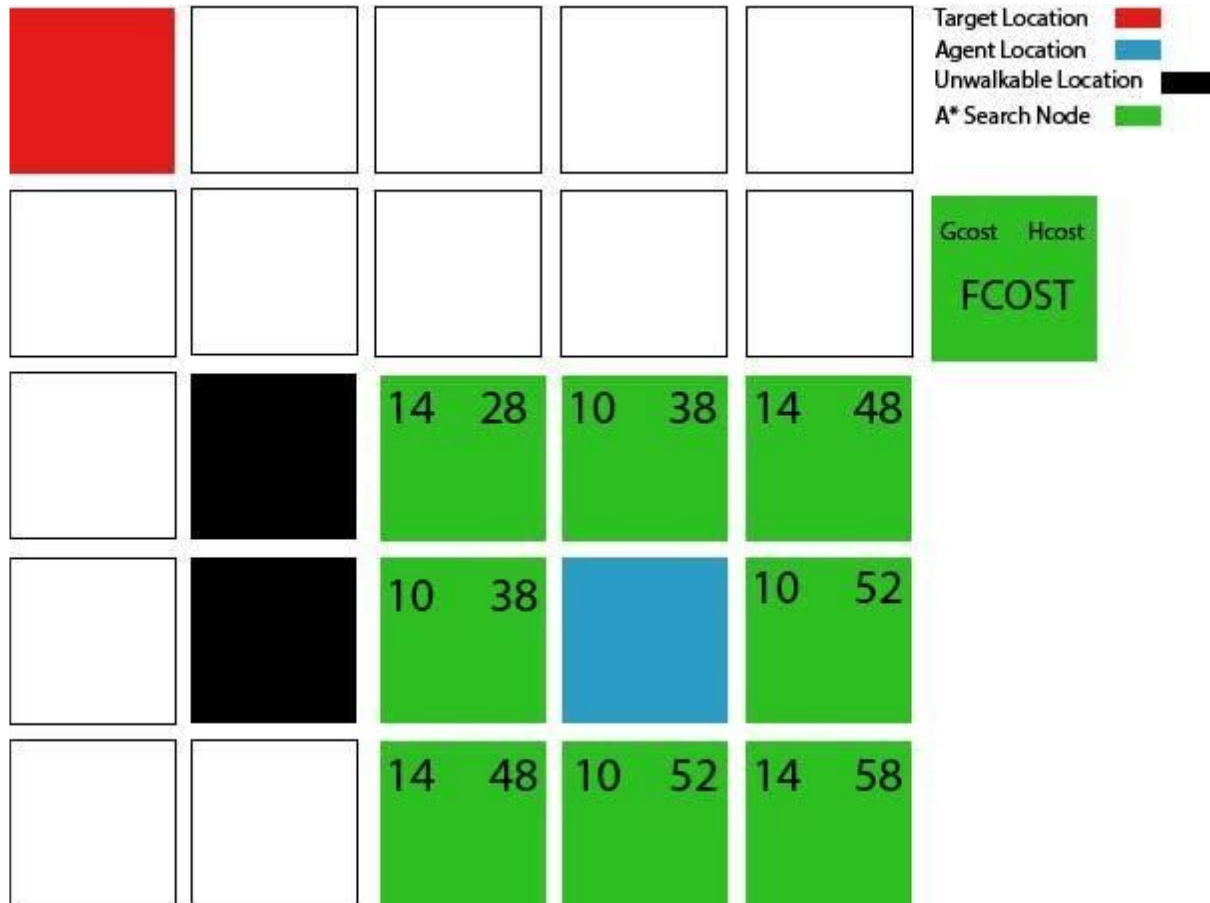
Charisma	<ul style="list-style-type: none"> - Improves the rate at which a resource is produced. - Within combat, charisma provides all fellow agents with a correlated probability for an additional turn to move
----------	---

2.5.2 Agent Pathfinding

Within Valence, agents navigate the game world through the use of a pathfinding algorithm known as A*. A* tasks agents to update their location on a game world graph composed of a suite of tiles (graph tile count can vary based on the level of detail a developer wishes to implement). Once a target location has been assigned to a particular agent, they will calculate a possible path given the location all known walkable and unwalkable areas (such as buildings, walls, etc.).

As seen in figure 3 below, the path calculation is determined through series of movement costs, F, G and H costs. The G cost is based on the surrounding node's distance from the agent. The H cost is based on the surrounding node's distance from the target. Both of these values are then substituted into a final calculation, referred to as the F cost. Once calculated, the agent will traverse the shortest distance to the target, based on the F cost of the agent's surrounding nodes.

Figure 3: Agent Path Calculation



All agents within *Valence* traverse the world through basic point to point targets. These locations can either be determined at random to simulate a wandering behaviour, or told explicitly when searching for a known destination (eg. a food source). Given the behaviour of this movement system, the A* algorithm provides the best solution by guaranteeing that all agent will follow the shortest path, each time a path is generated.

2.5.3 Dynamic Agent Needs

Agents within *Valence* each have a series of dynamically changing needs. These needs include:

Table 5 - Dynamic Agent Needs

Agent Need	Description
------------	-------------

Health	<ul style="list-style-type: none"> - Dictates overall agent well being. When this variable depletes the corresponding agent dies. - Health is reduced when damage is taken, if an agent is thirsty and unable to find water, or if their hunger depletes past an acceptable amount.
Morale	<ul style="list-style-type: none"> - Dictates the current happiness of an agent. - Agent productivity is directly related to their happiness. - Unhappiness can be produced through a variety of scenarios: <ul style="list-style-type: none"> - A fellow agent dies within a set social radius. - An agent is assigned to a role which does not suite their attribute bias. - An agent can't find food or water - Insufficient amount of power is being produced - An agent is surrounded by fellow unhappy agents
Energy	<ul style="list-style-type: none"> - Each agent has a unique energy level which dictates the amount of time they can work. - An agent whose energy has been depleted by working, will return back to their living quarters to re-energize.

2.5.4 Agent Social Physics

Valence utilizes agency to replicate how people interact with each other in various social scenarios. In order to better replicate the social exchange with real users, *Valence* captures these interactions and social context using a diverse suite of agent models and behaviours. These recorded interactions in turn, affect how agents perform within their environment. For instance, happy agents will socialize amongst themselves, resulting in cliques of motivated agents who perform their tasks more efficiently.

2.6 Gameplay Inspiration and References

2.6.1 - SimCity



The quintessential city builder game, our game draws heavily from the SimCity franchise. The build mode incorporates many fundamental aspects of city builders established in SimCity including interface, controls and visual information. Player interaction such as 'zoning' (the act of designating an area for a specific building type) are inspired directly from SimCity.

2.6.2 - Banished



Banished, a city building game set in the 19th century, tasks players to manage a settlement of exiles, resources and build a functioning community. *Banished* emphasizes on resource management and assigning agents to roles within the community, a mechanic which we believe works well within our game. As such our game also uses *Banished* as a point of reference for interface design for these particular elements.

2.6.3 - Fallout Shelter



Fallout Shelter, an idle game for iPhone and Android where players manage a post-apocalyptic nuclear bunker of Vault Dwellers. It features both resource and population management. Fallout's simple correlations between Vault Dweller Stats, Jobs and Resources made for a game that was easy to pick up and play.

2.6.4 - XCOM: Enemy Unknown



For the Exploration Mode we had two primary influences, the first of which is XCOM: Enemy Unknown. XCOM features a Turn-Based Strategy mode in which players move a squad of soldiers through a play area and attempt to capture a location or route from a series of enemies. XCOM's tactile interface and turn based structure are major influences for the Exploration Mode. Influences taken from XCOM specifically include the small scale skirmish design, stealth options and ranged combat focus.

2.6.5 - Fire Emblem



Fire Emblem represents the other primary influence on the Exploration Mode. Most of Fire Emblems actions are accomplished by merely moving units about the battlefield allowing for a tactical gameplay experience that is easy to pick up and

play, but still allows for a depth of tactical gameplay permutation in the way in which you order and command your units. It is this kind of design that we hope to incorporate into Valence.

3. Controls and Interface

3.1 Build Mode Controls

Build Mode controls and interface are inspired heavily by city builder genre conventions.

Table 4 - Build Mode Control Inputs

Input	Action
WASD or Arrow Keys	Move Camera
Alt + Mouse Drag	Rotate Camera
Left Mouse Button	Interact
Right Mouse Button	Inspect
Middle Mouse Wheel	Zoom Camera
Tab	Game Menu
ESC	Options Menu
Space	Toggle Live / Build UI
Number Keys	Quick Select Zone Type

3.2 Explore Mode Controls

The Explore mode controls build off the Build Mode controls, but are modified to suit the needs of combat scenarios, and have been influenced by turn-based games featuring tactical combat.

Table 5 - Explore Mode Control Inputs

Input	Action
WASD or Arrow Keys	Move Camera
Alt + Mouse Drag	Rotate Camera

Left Mouse Button	Interact
Right Mouse Button	Inspect
Middle Mouse Wheel	Zoom Camera
ESC	Options Menu
Tab	Game Menu
Enter	End Turn
Space	Cycle Selected Unit
Number Keys	Quick Select Unit

3.3 Icons

The following icons will be presented on unit cards as well as on the HUD. They will be simple vector art in black and white tones, allowing for quick understanding of various resources, currency, number of units, colony info, and settler attributes.

3.4 Interface

The interface is composed of the Heads Up Display (HUD) and the Information Panel.

The HUD covers the important information that a player needs to know at all times.

It includes:

- Time since start of play session
- The number of agents in the colony at a given time
- Scrap Amount
- Colony Hunger level (an average of each settler's hunger)
- Colony Thirst level (an average of each settler's thirst)
- Colony Morale level (an average of each settler's morale)

3.5 Interface References

3.5.1 Banished



Banished has windows which encompass inventories of units as well as individual structures. This is done by having consistent icons across the different layouts, so one can identify health (represented by hearts) for citizens as well as the entire town. *Valence* will have settlement information at the top, and settler attributes which need to have overlapping information which will be using the same icons.

3.5.2 Starcraft II



Starcraft II's unit portraits, as well as small UI considerations such as (but not limited to) health bars unit selection highlights, and resource HUD are adapted into *Valence* as well. These interface fields accurately summarize the entire levels to the player at a glance and without too much effort.

4. Game Environment



4.1 Metro Stations





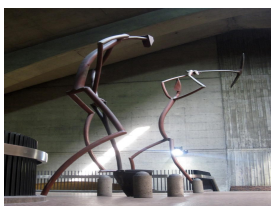
4.1.1 Main Station

The core build environment will be created as a fictional metro station located in the Montreal Metro that connects various real world stations. The core build area will incorporate common themes of a typical Montreal station. The build environment layout is essential for providing the player with a game world that allows them to create all of their buildings. The layout will assist with agent movement to allow them to perform their tasks without impeding them. Due to the fact that real world metros are smaller in size and don't facilitate large settlements, the build environment will be significantly larger.

4.1.2 Exploration Stations

Exploration Stations will be created based off of real world stations set in montreal. Stations have been chosen due to their unique landmarks such as statues and sculptures and will be incorporated into the designs to further push the sense of realism. The following list demonstrates which station will be recreated including their unique landmark.

Station	Artist	Artwork	Landmark
De le Savane	Maurice Lemieux	Calcite (1984)	
Namur	Pierre Granche	Système (1984)	

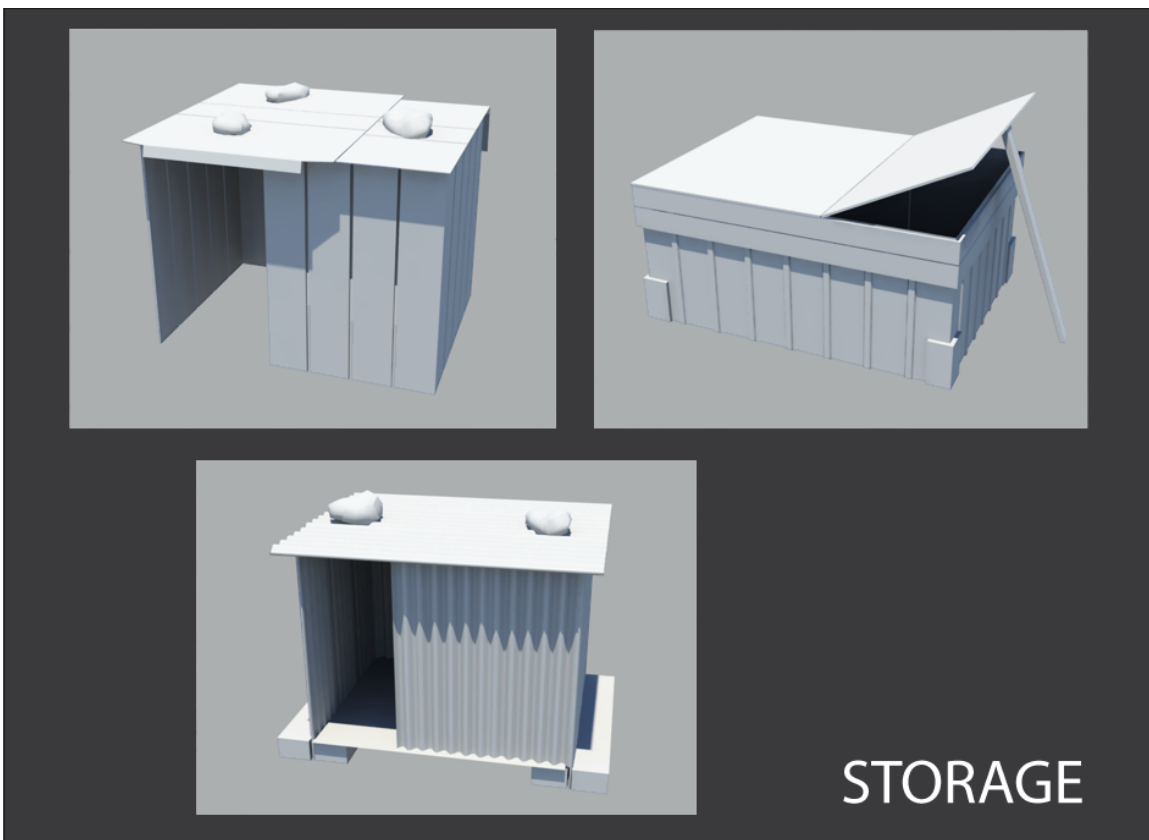
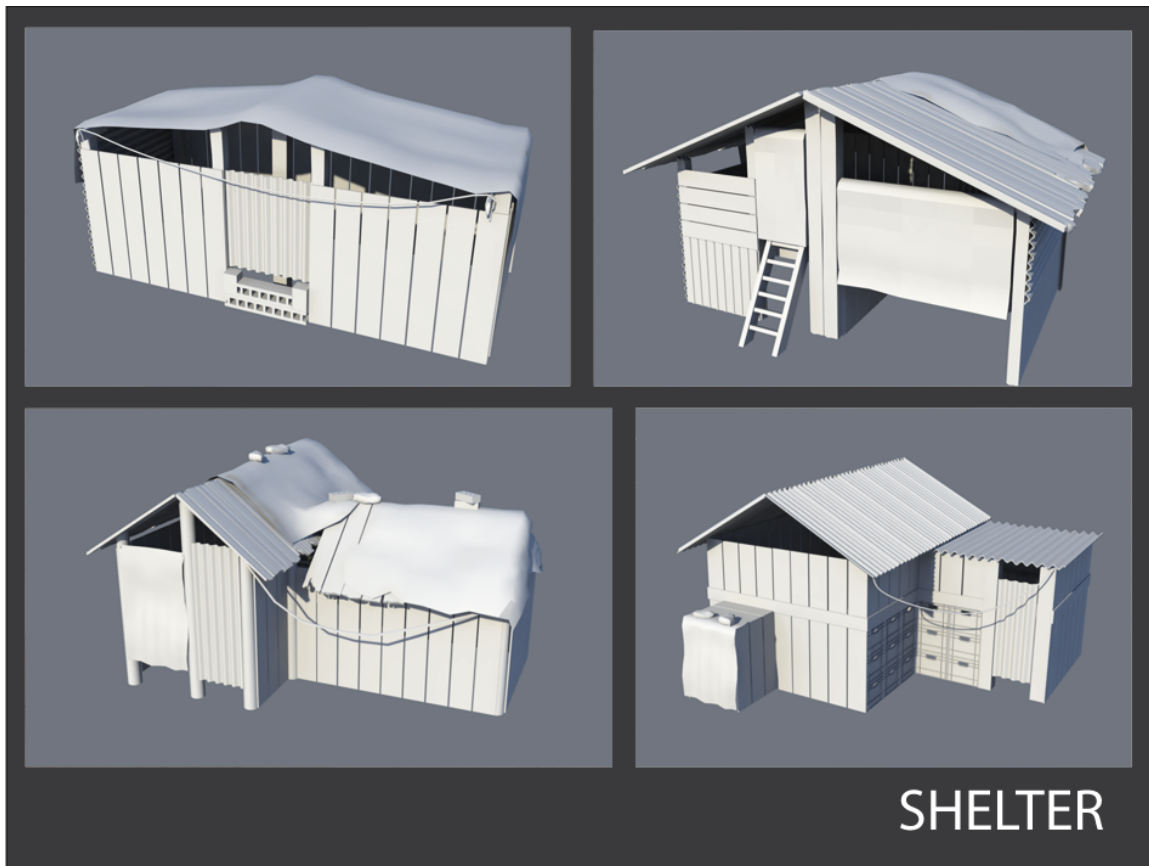
Place-Saint-Henri	Joseph-Arthur Vincent	Jacques Cartier (2001)	
Lionel-Groulx	Joseph Rifesser	The Tree of Life (1978)	
Georges-Vanier	Michel Dernuet	Un arbre dans le parc (1980)	
Berri-UQAM	Raoul Hunter	Monument à Mère Émilie Gamelin (2000)	
Monk	Germain Bergeron	Pic et Pelle (1978)	

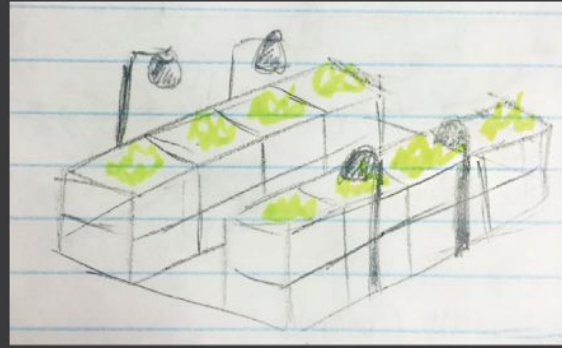
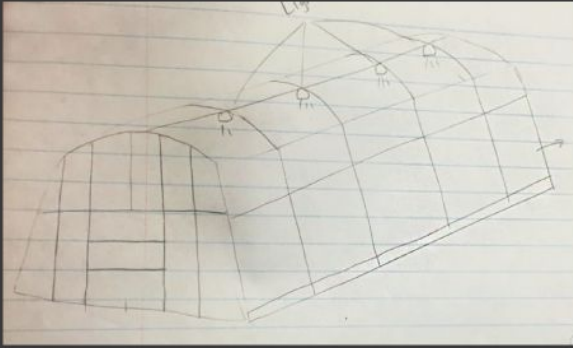
4.2 Buildings

Player made buildings are 3D models that are constructed onto designated build zones. These structures are separated into two categories: primary and secondary buildings. Primary buildings consist of an agent shelter, farm, storage, hospital, and the power station.

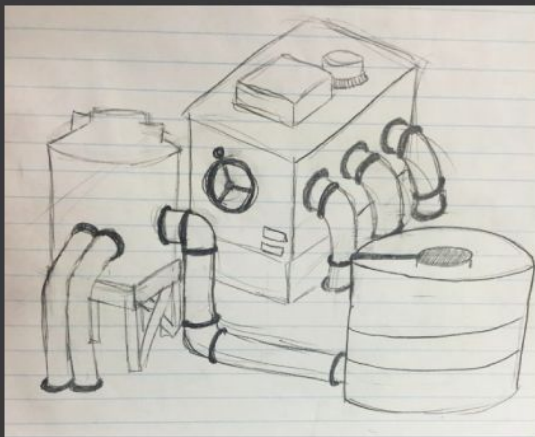
Secondary buildings consist of the shrine, tavern, and school. The agent shelter, farm and storage have multiple designs that the player cannot choose directly but provide variations to the building types. The others will have one design with strong characteristics that players can distinguish easily. Every building has a short

under-construction animation to provide the player with feedback. Below are the design of buildings.

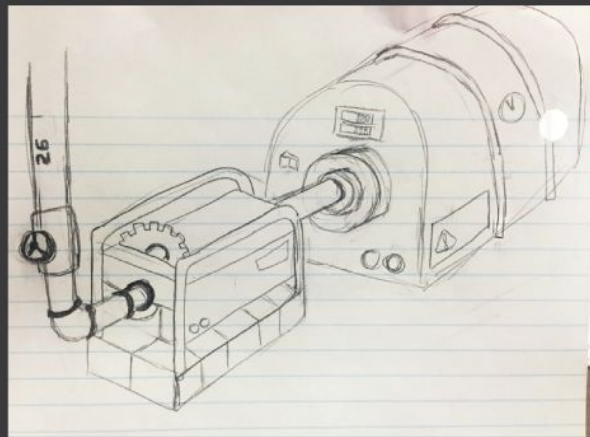




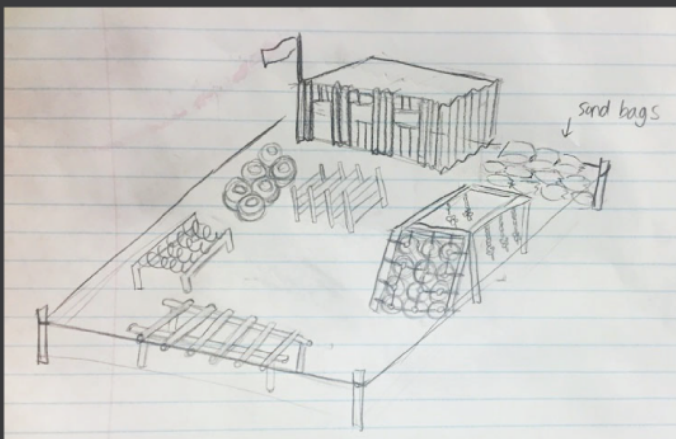
FARM



WATER STATION



POWER STATION



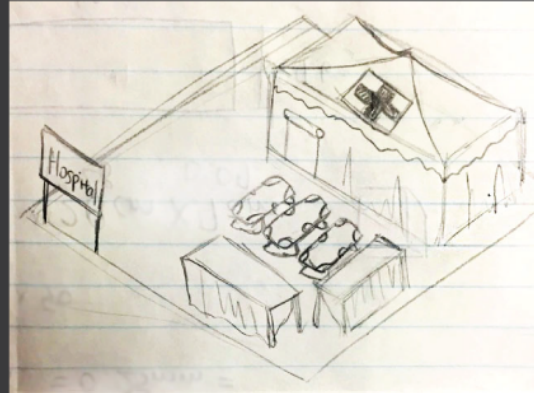
SCHOOL



SHRINE



TAVERN



HOSPITAL

4.3 Props

Props are 3D models that compose the game world as well as decorate the environment. Props have been segmented into two categories: Global and local props. Global props enclose the environment and allow for agent movement into other sections of the metro. Items such as walls, floors, pillars, garbage cans and benches are considered a global prop. Global props are geared towards environmental design whereas local props are geared towards building design.

Local props consist of objects that can be used to create buildings. There are a number of unique items that are considered local props but will only be used for specific building types. Local props will be generated and placed randomly within a selected zone. Generated objects will be related to a specific building type to provide a unique visual style.

5.0 Technology

5.1 Game Engine

Valence will be developed in Unity 3D (version 5.1.1). The game will leverage Unity's built in 3D rendering tools and physics engine.

5.2 Folk Generation

Folk characters will be generated using a python script. This is done to create the greatest number of unique folk characters in a short period of time. Having a wide range of unique characters, the player can differentiate them by the masses. Folk faction consist in 96 variations of characters in which it will be divided equally between male and female. To create the character generation, each folk character will be divided into three sections. Skin, clothes and hair. Skin will have two different variations which are light, and medium skin. Clothes will have four variations, containing two different models and two different textures. Hair will have six different variations, consisting of two different models and three different colors. By interchanging the different section, each gender will have 48 unique folk characters.

The folk generator will work by giving the developers a GUI box that will only contain one button. This button will call the main function in which the code will randomly select the gender, outfit, skin color and hair. The image shown below shows how the GUI generator works.

```
1 import maya.cmds as cmds
2 import random
3 import math
4
5 # =====#
6 # === User Interface ===#
7 # =====#
8 window = cmds.window (title = "Character Generator", menuBar = True, widthHeight = (1080, 720))
9
10 cmds.menu (label = "Basic Options")
11 cmds.menuItem (label = "New scene", command = ('cmds.file (new = True, force = True)'))
12 cmds.menuItem (label = "Delete selected", command = ('cmds.delete()'))
13
14 cmds.columnLayout()
15 cmds.button (label = "Create Agent", bgc = (175, 110, 13), command = ('createChar()'))
16
17 cmds.setParent ('..')
18 cmds.setParent ('..')
19 cmds.showWindow (window)
20
```

When executing the main function, it will call the skin, outfit and hair functions in order to retrieve the information and import the different components of the folk character. By calling each function, it will randomly select the element of each section and return the information to the main function. With the information

returned, the main function can import the selected element. Each component of the folk character will be imported separately since each will be in distinct fbx files. The image below shows how the main function calls each component of folk characters and imports the fbx files depending on the return value.

```

21 # =====#
22 # === Functions ===#
23 # =====#
24 def createChar():
25     cmds.file (f = True, new = True)
26     cmds.modelEditor (displayAppearance = 'smoothShaded')
27
28     #cmds.cameraView (camera = persp)
29
30     # Path of File
31     pathOfFile = 'C:\\Users\\'
32     user = 'Naydug'
33     pathOfFile = pathOfFile + user + '\\Desktop\\Agent_Parts\\'
34
35     # Random generation
36     gender = random.randrange (1, 3)
37
38     if gender == 1:
39         genderSelect = 'male'
40         print 'Gender: Male'
41
42     elif gender == 2:
43         genderSelect = 'female'
44         print 'Gender: Female'
45
46     # Calling functions
47     skinCol = agentSkin ()
48     clothCol = agentCloth ()
49     hairCol = agentHair ()
50
51     # Import Skin
52     skinImport = pathOfFile + genderSelect + '_' + skinCol + '.fbx'
53     print skinImport
54     cmds.file (skinImport, i = True)
55
56     # Import Clothes
57     clothImport = pathOfFile + genderSelect + '_' + clothCol + '.fbx'
58     cmds.file (clothImport, i = True)
59
60     # Import Hair
61     hairImport = pathOfFile + genderSelect + '_' + hairCol + '.fbx'
62     cmds.file (hairImport, i = True)
63
64 # -----#

```

5.3 Procedural Prop Placement

In order to create game environment that look natural and a believable we will develop a tool in Unity that will dynamically place game assets in the environment in varied ways. When a player places a building a primary building of that type is placed within a specified space. Then 'world props' (which is to say environment objects that act as set dressing) are place within the space with physics collisions enabled. If an object comes to rest outside of the space or is bisecting another object it is deleted.

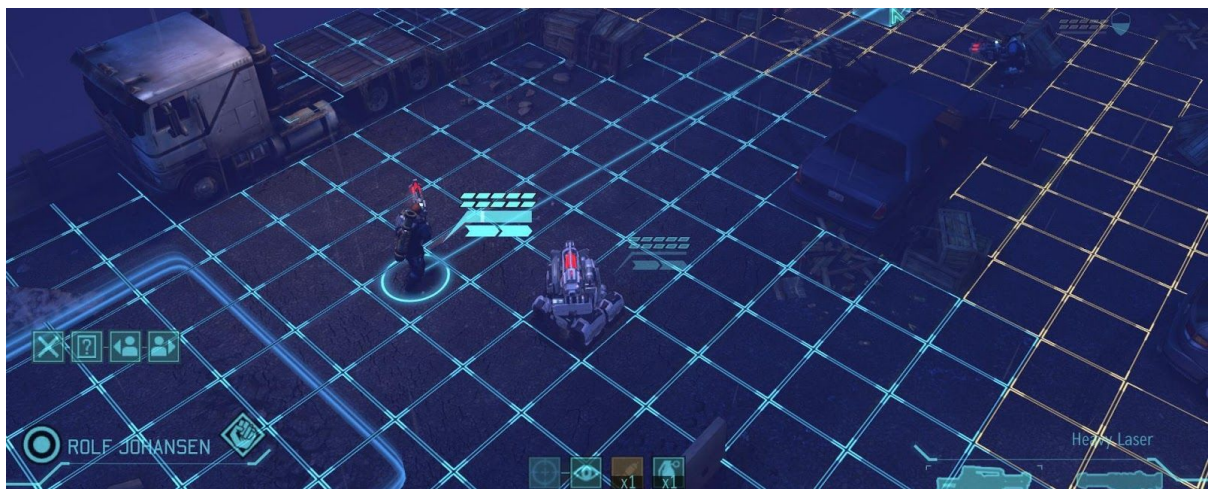
6.0 Style

6.1 Art

The style of *Valence* is realistic since we wanted to portray a game in which the player would believe in the circumstances the characters are in. We also took games with a realistic style as an inspiration like XCOM and Metro 2033. *Valence* has a dark and grungy look and feel since we wanted to portray that agents past through a post -apocalypse period. Having a dark look to our game, we also can link the characters and the environment with the backstory.

6.2 Art Influences

6.2.1 XCOM



Reference Image 1: XCOM's grid in an isometric view [1]

Valence utilizes XCOM: Enemy Unknown as an influence for the exploration stage. Similar to XCOM, a grid is displayed so the player can better view where the agents are positioned. Given the similarities in game design, our team created a similar strategic camera perspective. By placing it in a isometric view the player can better observe of the overall environment and plan their next combat move.

In addition, the exploration interface is further inspired by XCOM. This was implemented by having valuable information for the player displayed on the screen and additional information when the player clicks on an agent or a building.

6.2.2 Metro 2033



Reference Image 2: A character from Metro: 2033 [2]

Metro 2033 is a major influence for the art style and plot featured in *Valence*. Taking place in a metro station and in a post-apocalyptic period, *Valence* also has the same environment and setting. In addition, the style for our folk characters was also further influenced by the game. Elements like gas masks and rugged outfits have been implemented in *Valence* to portray the same look and feel as metro 2033. Similarly, buildings also have a used and dilapidated look, built using resources and material scavenged around the environment.

6.3 Sound

The goal of sound design is to compliment the dark and gritty visual feel of the game while augmenting that feeling to the player.

Besides audio tracks playing in background, ambient noise is added in the form of white noise, which is created through string and woodwind instruments. It contains frequencies which are audible to the human hearing range in multiple ranges and is played through higher and lower pitches in lethargic tones. Pink noise - which are

synths with reduced volume at each octave - are also added alongside white noise to amplify the ambience.

There will be sound effects for clicks on the button, as well as voice-overs for social interactions between the folk, to create add immersion the game.

6.4 Sound References

6.4.1 Grow Home



The background tracks will be spaced out to give the player a feeling of emptiness and take in the spatial vastness of the game. Grow Home, a game all about ambient music lets the music fade away to give the sound effects some breathing room and take the spotlight until the next track starts out.

Reference clip: <https://youtu.be/wCaI0w67n-Q?t=2m13s>

6.4.2 Pandemic 2



Pandemic 2 switches through different levels of tension to help create music of different parts of the game. These levels essentially dictate how “intense” an audio clip is. It can include (but is not limited to) increased BPM, more brassy and orchestral sounds, alternating between loops to give a different sound, and even introducing metals drums or alarm sounds at irregular lengths.

Valence’s audio tracks change in tension levels, going back and forth to allow for an increase and decrease in levels. Accordingly, tracks of different beats-per-minute can be switched in and out without making the track come to an abrupt stop.

Reference Clip: <http://www.crazymonkeygames.com/Pandemic-2.html>

6.4.3 Superhot

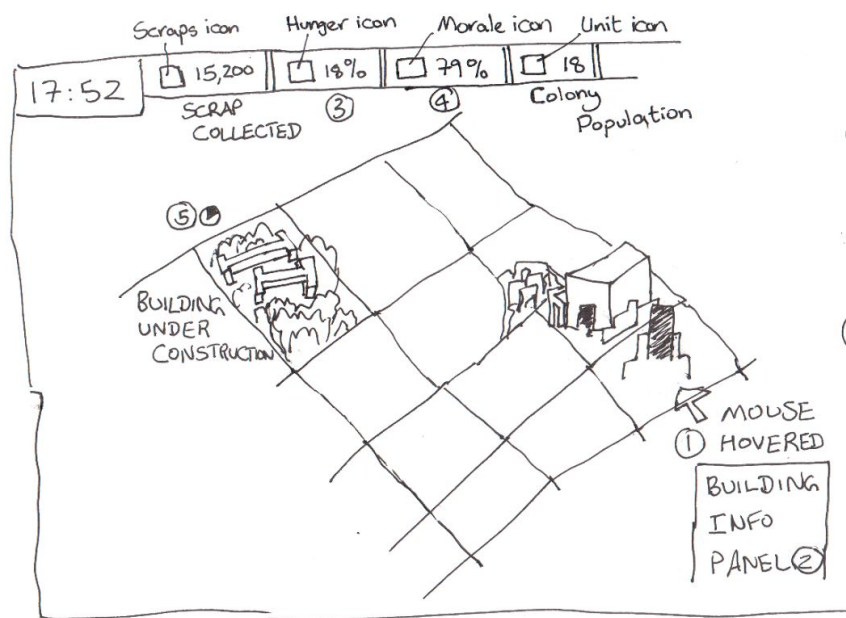


Superhot has one of the closest matching tones in terms of eeriness which is forced to change in tempo based on the timing of the game. As *Valence* will have an Exploration mode, as well as a Build mode, the tempo of the game will have to change based on what the player decides to do, and have to create that transition seamlessly.

Reference Clip: <https://www.youtube.com/watch?v=oShPbvoD50Q>



Appendix

- [1] [XCOM's grid](#)
- [2] [Metro: 2033 Character](#)
- [3] [Brook's AI Lab](#)
- [4] [Brook's Introduction to Subsumption](#)
- [5] [Real world application of complex systems](#)
- [6] [Guide to procedural learning systems](#)



- ③ Global hunger level.
Lower = better
- ④ Global morale percentage.
Higher = better
- ⑤ Pie chart showing
time till building
finishes construction

LIVE VIEW BUILDINGS

- ① Icon for mouse changes when hovering from  to , which is the shape of the building.
- ② Includes amount of scrap gained through demolition, as well as info on how the building affects the colony. E.g. +5% Morale boost.



MOUSE
CLICKED



	Name	Dave
	Role	Farmer
Gender	♂	
Health	82/100	
Stamina	70/100	
Hunger	5%	
Morale	88%	



MOUSE
CLICKED



	Name	Dave
	Role	Farmer
Gender	♂	
Health	82/100	
Stamina	70/100	
Hunger	5%	
Morale	88%	

	Name	Dave
	Role	Farmer
Gender	♂	
Health	82/100	
Stamina	70/100	
Hunger	5%	
Morale	88%	
Attributes		
Strength	5	
Intelligence	1	
Agility	2	
Perception	3	
Inventory		
Details		

	Name	Dave	
	Role	Farmer	
90% Full			
<input checked="" type="checkbox"/> Tomato	9		
<input checked="" type="checkbox"/> Cabbage	2		
<input checked="" type="checkbox"/> Scrap	17		
Inventory			
Details			